

Introduction to the CHI Algorithms

Phil Hawkes, Cameron McDonald

Cryptographic Hash Initiative

Qualcomm Product Security Initiative

{phawkes,cameronm}@qualcomm.com

Website: www.qualcomm.com.au/CHI.html

Overview



	Little CHI	Big CHI
	CHI-224/256	CHI-384/512
State size (bits)	384	544
Msg Block (bits)	512	1024
Num Steps	20	40
Operations	64-bit ADD & Logic Ops	
	64-bit Shift, ROTR64, SWAP32	
	SWAP8 (<i>byteswap</i>)	
	DROTR32	

Boring stuff

- Variant of Merkle-Damgard Chaining
 - Prevent length extension
 - Final block state words rotated by one bit prior to processing
- Variant of Davies-Meyer
 - Prevent Fixed Points
 - Rotate input hash words before XOR with block cipher output
 - Security Pseudo-proof

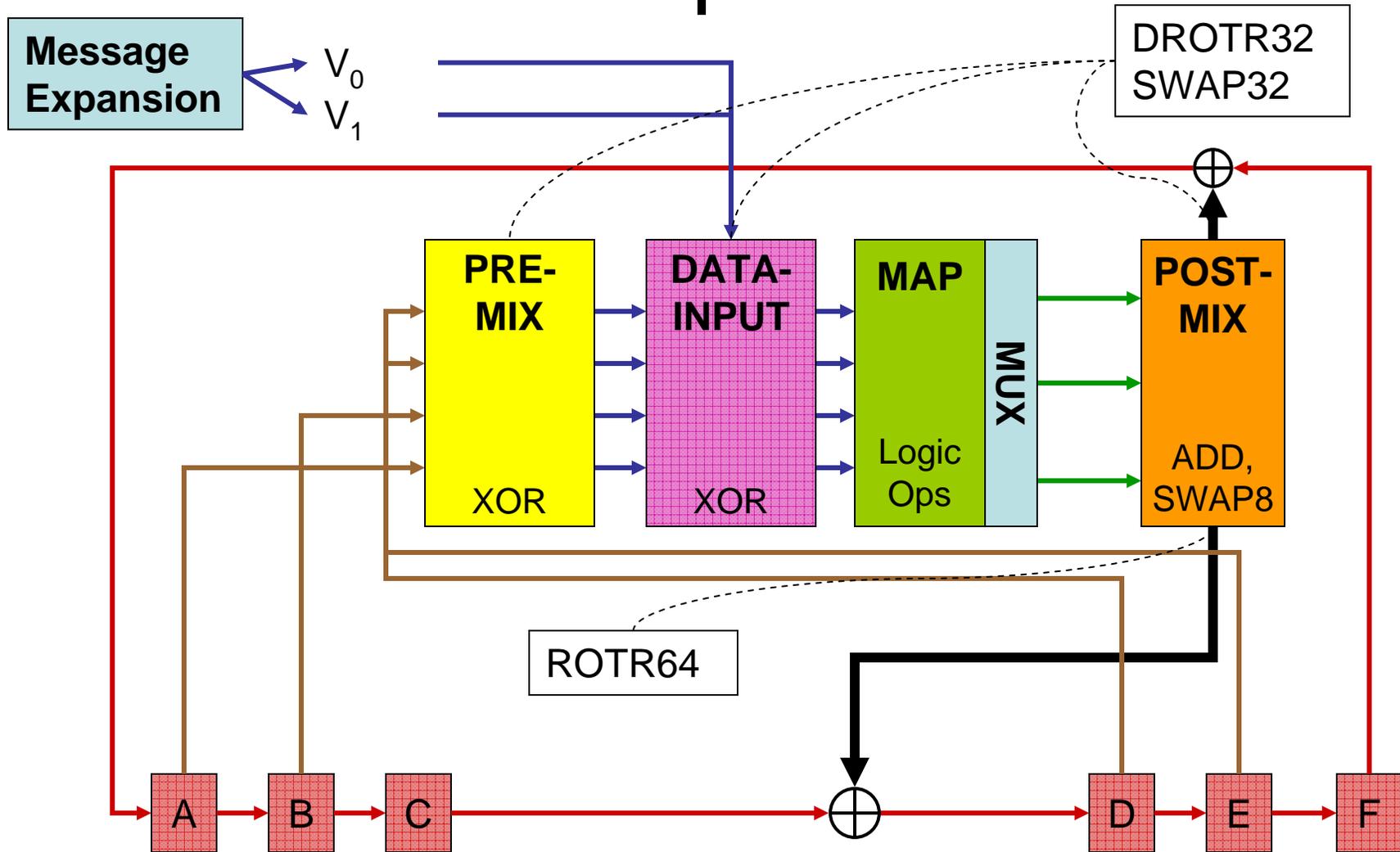
Differential paths

- Deterministic steps
 - 1st few steps while combining message block
 - Attacker forms message block pair conforming to conditions (message modification)
 - Designer wants lots of conditions
- Probabilistic steps
 - Remaining steps
 - Attacker “hopes” differential path holds
 - Designer wants low probabilities
- Need Good Nonlinear functions & Diffusion

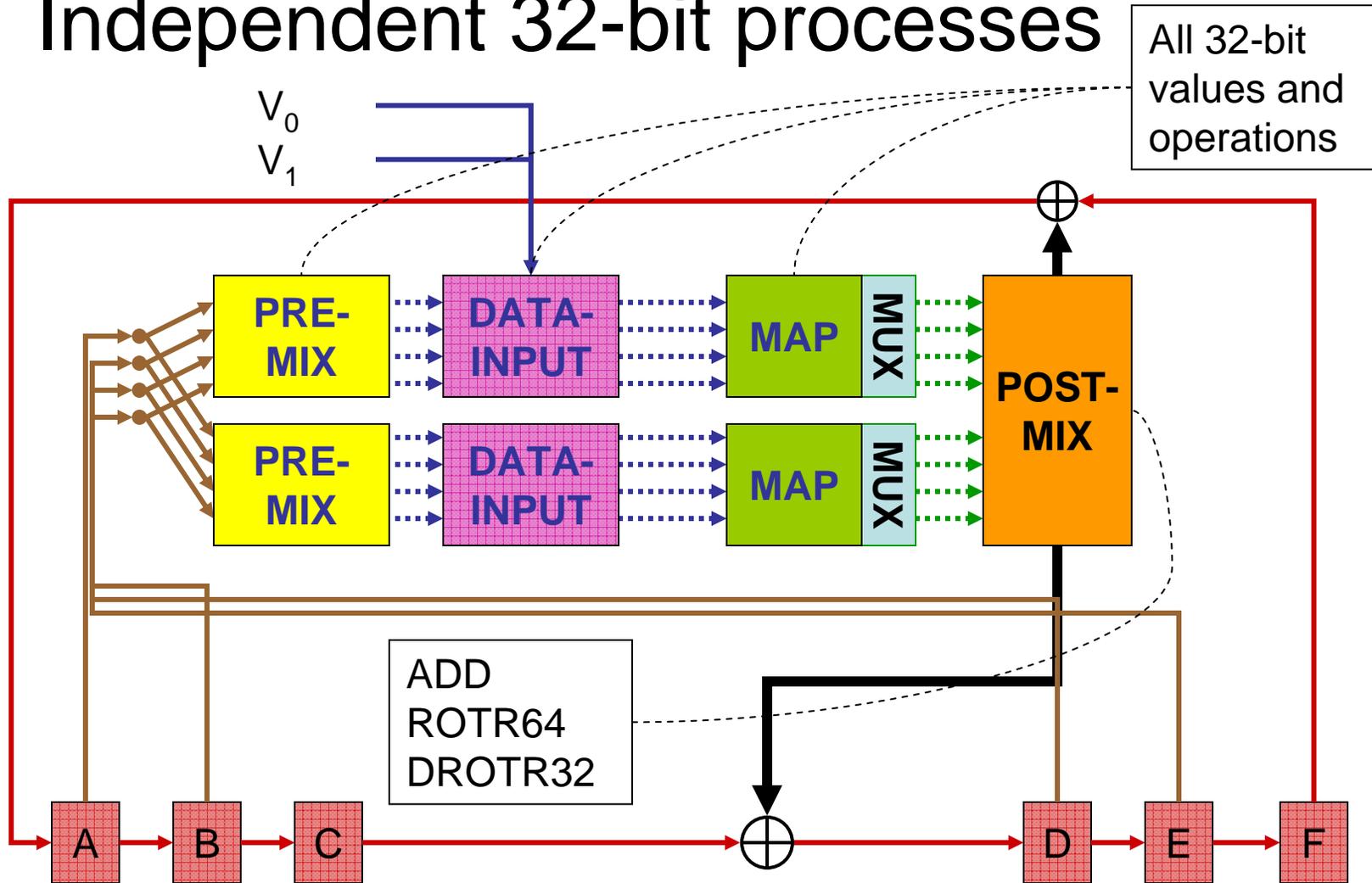
CHI Design Philosophy

- Aim: Good efficiency on all platforms
- Focus on Step function.
 - ADD and bit-sliced fn: Orthogonal nonlinearity
 - MAP: 4-input 3-output bit-sliced fn:
 - Excellent differential properties, Reduce ADD
 - Diffusion: Multiple rotations, single SWAP8
 - Little CHI: mostly two parallel 32-bit processes
- Msg expansion similar to SHA-2 but linear
 - Replaced ADD with XOR, Easier to analyze
 - Cheaper in Hardware

Little CHI Step Function



Little CHI: Independent 32-bit processes





CHI MAP

- Balance “crypto” properties & efficiency
- “Crypto” properties (accounting for ADD)
 - Focus on Single bit input XOR diffs:
 - No zero output difference
 - Distribution skewed: more likely to get more output diffs
 - Signed (+/-) bit differences
 - Almost flat distribution
 - Almost doubles number of conditions (Deterministic steps)
 - Good algebraic properties
- Efficiency: Long chain of operations in parallel
 - 15 logical instructions
 - Very efficient in HW (pipeline). Efficient for all SW.

Diffusion

- Slice = single bit position
 - *MAP* 4 inputs & 3 outputs in a slice are related
- **PRE-MIXING** & **POST-MIXING**
 - Ensure mixing between slices
 - Expand recent variables by factor of 3
 - Parent slice: slice in prev step contributes to this slice
 - Any two slices share at most one parent slice
- **DATA-INPUT**
 - Complicate cancelling input diffs in *MAP*
 - Expand input word by factor of 4, XOR in
- [Link to Little CHI](#), [Link to Big CHI](#)

Results so far

Metric		SHA			CHI	
		1	256	512	Little	Big
1-bit Local Collision	Number of Conditions	4	39		91*	147*
	Probability	2 ⁻⁴	2 ⁻³⁹		2 ^{-44*}	2 ^{-74*}
Min weight Disturbance vector (so far)	Num Words	80 [#]	64 [#]	80	40	80
	All steps	44	?	See <i>Big CHI</i>	266	834
	Prob Steps**	25	?		139	474

* Two steps of difference pattern. No local collision known

**Steps where differential is modeled probabilistically

32-bit words

Efficiency

Equivalent 64-bit ops	224/256		384/512	
	SHA-2	CHI	SHA-2	CHI
ADD	300	40	760	120
Logic Ops	512	826	1296	1809
ROTR64	-	166	736	1080
DROT32	240	280	-	-
SHIFT	48	80	128	200
SWAP32	-	100	-	40
SWAP8	-	20	-	40

Current Implementations

		Little		Big	
		CHI	SHA-2	CHI	SHA-2
SW cycles/byte	32-bit	49	~22	78	41-117
	64-bit	24	~20	16	~13
RAM (bytes)		198	140	318	280
HW	Size (kGE)	~20	18	←CAST Inc	
	MB/s	600	484		
	Clock (MHz)	188	500		
	μm	0.13	0.09		

Conclusion

- Good efficiency on all platforms
 - Significantly improved for HW
- Design builds on existing analysis techniques
- Bit-slice *MAP*
 - Excellent nonlinear differential properties
- *PRE-MIXING & POST-MIXING*
 - Diffusion between slices
- *DATA-INPUT*
 - Diffusion prevents easy cancellation of bit differences
- Website: www.qualcomm.com.au/CHI.html

Cryptographic Hash Initiative

Part of the Qualcomm Product Security Initiative headed by Greg Rose

Design Team

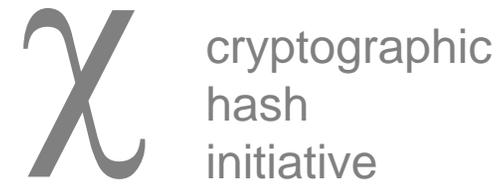
- Leads: Phil Hawkes, Cameron McDonald
- Major contributors: Brian Rosenberg, Lu Xiao
- Notable contributors: David Jacobson, Steve Millendorf,
Craig Northway, Yafei Yang

Implementation

- Software: Cameron McDonald (Lead), Craig Brown,
Craig Northway, Jessica Purser
- Hardware: Bijan Ansari

Thanks also to:

- Arun Balakrishnan, Alexander Gantman, John Jozwiak, Yinian Mao,
Michael Paddon, Anand Palanigounder, Aram Perez and
Miriam Wiggers de Vries



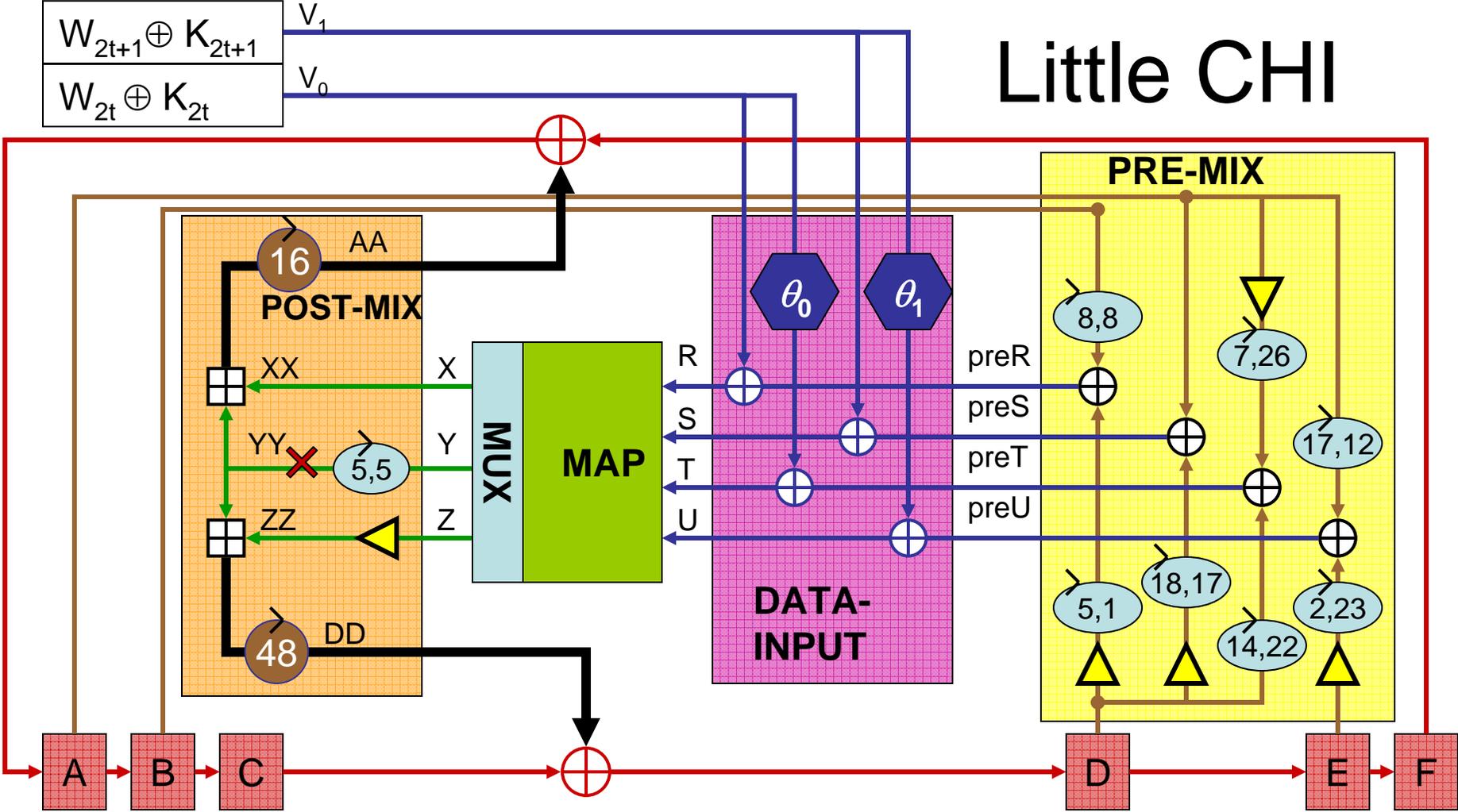
Backup Slides

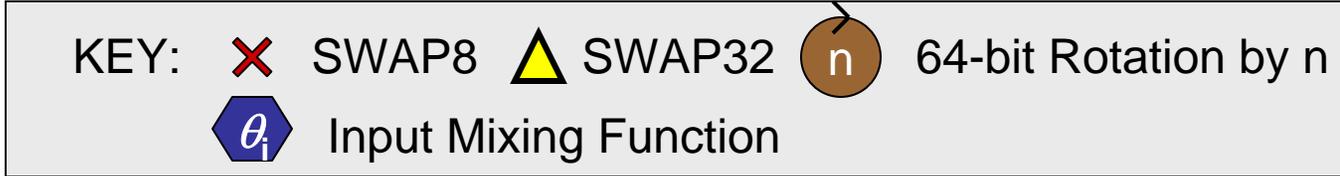
Details of Step Functions

KEY:  SWAP8  SWAP32  Rotate upper 32-bit half by ru
 Rotate lower 32-bit half by rl

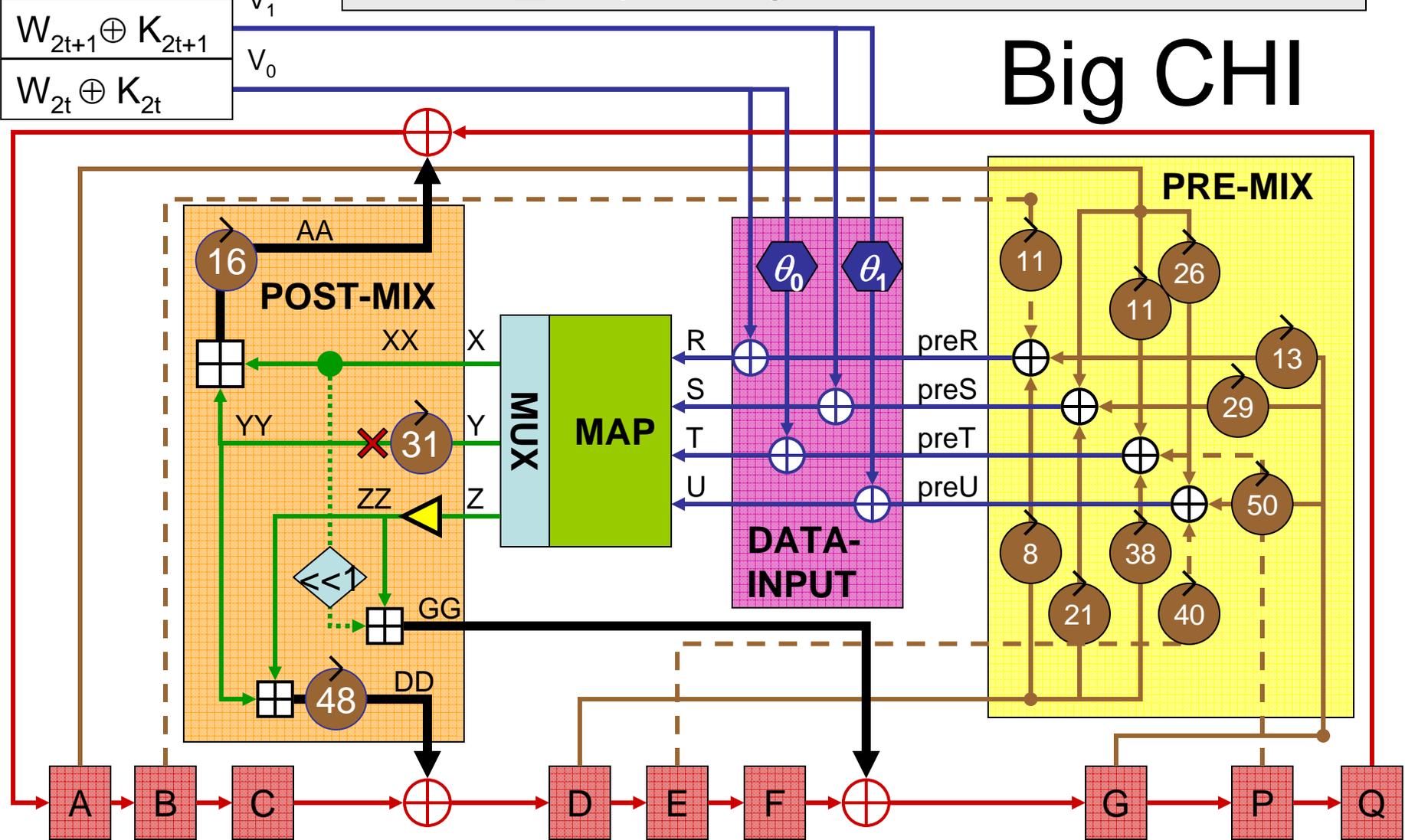
 θ_i Input Mixing Function  64-bit Rotation right by n

Little CHI





Big CHI



DATA-INPUT Diffusion

- $\theta_0^{\{256\}}(x) = \text{DROTR32}^{(21,21)}(x) \oplus$
 $\text{DROTR32}^{(26,26)}(x) \oplus$
 $\text{DROTR32}^{(30,30)}(x)$
- $\theta_1^{\{256\}}(x) = \text{DROTR32}^{(1,1)}(x) \oplus$
 $\text{DROTR32}^{(15,15)}(x) \oplus$
 $\text{DROTR32}^{(25,25)}(x)$
- $\theta_0^{\{512\}}(x) = \text{ROTR64}^{(5)}(x) \oplus \text{ROTR64}^{(6)}(x) \oplus$
 $\text{ROTR64}^{(43)}(x)$
- $\theta_1^{\{512\}}(x) = \text{ROTR64}^{(20)}(x) \oplus \text{ROTR64}^{(30)}(x) \oplus$
 $\text{ROTR64}^{(49)}(x)$